ASP.NET Themes and Skins

A theme is a collection of property settings that allow you to define the look of pages and controls, and then apply the look consistently across pages in a Web application, across an entire Web application, or across all Web applications on a server.

This overview contains the following sections:

Themes and Control Skins

Themes are made up of a set of elements: skins, cascading style sheets (CSS), images, and other resources. At a minimum, a theme will contain skins. Themes are defined in special directories in your Web site or on your Web server.

Skins

A skin file has the file name extension .skin and contains property settings for individual controls such as Button, Label, TextBox, or Calendar controls. Control skin settings are like the control markup itself, but contain only the properties you want to set as part of the theme. For example, the following is a control skin for a Button control:

```
<asp:button runat="server" BackColor="lightblue" ForeColor="black" />
```

You create .skin files in the Theme folder. A .skin file can contain one or more control skins for one or more control types. You can define skins in a separate file for each control or define all the skins for a theme in a single file.

There are two types of control skins, def*ault skins* and *named skins*:

- A default skin automatically applies to all controls of the same type when a theme is applied to a page. A control skin is a default skin if it does not have a **SkinID** attribute. For example, if you create a default skin for a Calendar control, the control skin applies to all Calendar controls on pages that use the theme. (Default skins are matched exactly by control type, so that a Button control skin applies to all Button controls, but not to LinkButton controls or to controls that derive from the Button object.)
- A named skin is a control skin with a SkinID property set. Named skins do not automatically apply to controls by type. Instead, you explicitly apply a named skin to a control by setting the control's SkinID property. Creating named skins allows you to set different skins for different instances of the same control in an application.

Cascading Style Sheets

A theme can also include a cascading style sheet (.css file). When you put a .css file in the theme folder, the style sheet is applied automatically as part of the theme. You define a style sheet using the file name extension .css in the theme folder.

Theme Graphics and Other Resources

Themes can also include graphics and other resources, such as script files or sound files. For example, part of your page theme might include a skin for a TreeView control. As part of the theme, you can include the graphics used to represent the expand button and the collapse button.

Typically, the resource files for the theme are in the same folder as the skin files for that theme, but they can be elsewhere in the Web application, in a subfolder of the theme folder for example. To refer to a resource file in a subfolder of the theme folder, use a path like the one shown in this Image control skin:

```
<asp:Image runat="server" ImageUrl="ThemeSubfolder/filename.ext" />
```

You can also store your resource files outside the theme folder. If you use the tilde (~) syntax to refer to the resource files, the Web application will automatically find the images. For example, if you place the resources for a theme in a subfolder of your application, you can use paths of the form *~/SubFolder/filename.ext* to refer to resource files, as in the following example.

```
<asp:Image runat="server" ImageUrl="~/AppSubfolder/filename.ext" />
```

Back to top

Scoping Themes

You can define themes for a single Web application, or as global themes that can be used by all applications on a Web server. After a theme is defined, it can be placed on individual pages using the **Theme** or **StyleSheetTheme** attribute of the @ Page directive, or it can be applied to all pages in an application by setting the <pages> element in the application configuration file. If the <pages> element is defined in the Machine.config file, the theme will apply to all pages in Web applications on the server.

Page Themes

A page theme is a theme folder with control skins, style sheets, graphics files and other resources created as a subfolder of the \App_Themes folder in your Web site. Each theme is a different subfolder of the \App_Themes folder. The following example shows a typical page theme, defining two themes named `BlueTheme` and `PinkTheme`.

```
MyWebSite
  App_Themes
    BlueTheme
      Controls.skin
      BlueTheme.css
    PinkTheme
      Controls.skin
      PinkTheme.css
```

Global Themes

A global theme is a theme that you can apply to all the Web sites on a server. Global themes allow you to define an overall look for your domain when you maintain multiple Web sites on the same server.

Global themes are like page themes in that they include property settings, style sheet settings, and graphics. However, global themes are stored in a folder named Themes that is global to the Web server. Any Web site on the server, and any page in any Web site, can reference a global theme.

Back to top

Theme Settings Precedence

You can specify the precedence that theme settings take over local control settings by specifying how the theme is applied.

If you set a page's Theme property, control settings in the theme and the page are merged to form the final settings for the control. If a control setting is defined in both the control and the theme, the control settings from the theme override any page settings on the control. This strategy enables the theme to create a consistent look across pages, even if controls on the pages already have individual property settings. For example, it allows you to apply a theme to a page you created in an earlier version of ASP.NET.

Alternatively, you can apply a theme as a style sheet theme by setting the page's StyleSheetTheme property. In this case, local page settings take precedence over those defined in the theme when the setting is defined in both places. This is the model used by cascading style sheets. You might apply a theme as a style sheet theme if you want to be able to set the properties of individual controls on the page while still applying a theme for an overall look.

Global theme elements cannot be partially replaced by elements of application-level themes. If you create an application-level theme with the same name as a global theme, theme elements in the application-level theme will not override the global theme elements.

Back to top

Properties You Can Define Using Themes

As a rule, you can use themes to define properties that concern a page or control's appearance or static content. You can set only those properties that have a ThemeableAttribute attribute set to **true** in the control class.

Properties that explicitly specify control behavior rather than appearance do not accept theme values. For example, you cannot set a Button control's CommandName property by using a theme. Similarly, you cannot use a theme to set a GridView control's AllowPaging property or DataSource property.

Note that you cannot use expression builders, which generate code expressions for assignment in a page at compile time, in themes or skins.

Back to top

Themes vs. Cascading Style Sheets

Themes are similar to cascading style sheets in that both themes and style sheets define a set of common attributes that can be applied to any page. However, themes differ from style sheets in the following ways:

- Themes can define many properties of a control or page, not just style properties. For example, using themes, you can specify the graphics for a TreeView control, the template layout of a GridView control, and so on.
- Themes can include graphics.
- Themes do not cascade the way style sheets do. By default, any property values defined in a theme referenced by a page's Theme property override the property values declaratively set on a control, unless you explicitly apply the theme using the StyleSheetTheme property. For more information, see the Theme Settings Precedence section above.
- Only one theme can be applied to each page. You cannot apply multiple themes to a page, unlike style sheets where multiple style sheets can be applied.

Back to top

Security Considerations

Themes can cause security issues when they are used on your Web site. Malicious themes can be used to:

- Alter a control's behavior so that it does not behave as expected.
- Inject client-side script, therefore posing a cross-site scripting risk.
- Alter validation.
- Expose sensitive information.
- The mitigations for these common threats are:

- Protect the global and application theme directories with proper access control settings. Only trusted users should be allowed to write files to the theme directories.
- Do not use themes from an untrusted source. Always examine any themes from outside your organization for malicious code before using them on you Web site.
- Do not expose the theme name in query data. Malicious users could use this information to use themes that are unknown to the developer and thereby expose sensitive information.

Back to top

Related Topics

How to: Define ASP.NET Page Themes
: Provides step-by-step instruction on how to apply themes to a page, a Web site, or globally.

How to: Apply ASP.NET Themes
: Provides step-by-step instruction on

How to: Disable ASP.NET Themes
: Provides step-by-step instruction on how to configure a page or control to ignore themes.

How to: Apply ASP.NET Themes Programmatically
: Provides step-by-step instruction on how to set page themes and style sheet themes programmatically.

Walkthrough: Creating User-Selectable Themes
: Provides a step-by-step tutorial that shows how to create an ASP.NET page that lets the user select a theme for the page.

ASP.NET Master Pages
: Describes how to create a consistent layout for pages in your Web site.

ASP.NET Web Server Controls and CSS Styles
: Describes how to work with CSS styles in ASP.NET Web pages.

ASP.NET Web Forms Pages
: Provides links to topics on creating ASP.NET Web pages.

Back to top

Reference

System.Web.UI.PageTheme